

Intelligent Document Processing

Automate Business with Fluid Workflow

Ting XU*, Yongmian ZHANG*, Xiaoqi WU*, and Wei MING*

Abstract

Form-like documents are essential in enterprise daily operations. Automatic information extraction from these documents is critical to business efficiency and unlocking insights from the form data. Manually created templates are routinely used to guide data extraction from a particular form type. This process can be cumbersome and time-consuming for end-users. To streamline the process, template-free solutions developed using fully data-driven approaches emerge in the market. However, they can be brittle when form structural and layout deviates from their training samples, which happens frequently in practice. To balance convenience and robustness, we introduce aiDocuDroid, a template-based information extraction solution that can handle a large variety of forms with varying structures and layouts with high accuracy. It requires minimal user effort to build form templates. aiDocuDroid is offered as both an on-premise solution and a cloud service, supporting printed or handwritten texts in English and Japanese.

1 Introduction

Every enterprise uses form-like documents. Invoices, purchase orders, tax forms, insurance quotes, and enrollment forms are common in daily business workflows. They are often scanned into raster images for digital archiving and transmission. Retrieving structured information from these documents often demands manual data entry, which is tedious and unscalable. To improve efficiency and unlock business insights, automatic extraction from form data has a great demand. The market size of Optical Character Recognition (OCR), one of the key technologies used in form processing, is expected to reach USD 13.38 billion by 2025¹⁾.

Information extraction (IE) is traditionally driven by manually created form templates, which prescribe the location and association of data for a particular form type. With advancement of Artificial Intelligence and machine learning, purely data-driven solutions aims at streamlining IE, eliminating human involvement as much as possible. In particular, Amazon, Microsoft and Google utilize their access to large-scale document datasets and have developed IE solutions requiring no form templates. Our experiment shows that these solution often struggle on semi-structured forms, which account for about 80% of the documents in business²⁾. Moreover, template-free approaches cannot easily re-key fields while re-keying is required by about 60% of the business documents²⁾. These limitations prevent their solutions from wide use in practice. In comparison, template-based solutions such as DocParser and ABBYY FlexiCapture make less errors in extracting forms with varying structure, but at the expense of significant user effort to build form templates. At KMLUS, we strike a balance between robustness and convenience, by keeping using form templates from which keywords and structures are detected automatically.

* Konica Minolta Laboratory USA, Konica Minolta Business Solutions USA, San Mateo, CA, USA

2 Challenges in form parsing

Information extraction from forms is more difficult from digitizing book pages, as the former needs to retrieve content relationships in addition to text detection and recognition.

Traditional OCR software, such as Tesseract³⁾, can successfully extract contents from articles and book pages, where contents are presented linearly without any association or linkage. In contrast, form documents often have key-value pairs, tables, or groups of options. The relationships may be conveyed spatially, through alignment, indentation, margin and ruling lines. Key texts can be distinctive from its associated contents through visual styles such as font, color, or shading. As the image and textual information intertwine, developing a form processing system often needs to utilize both Computer Vision (CV) and Natural Language Processing (NLP) techniques⁴⁾.

In addition to bearing relationships among document entities, forms of the same type may have different layout because some regions can expand, e.g., a table can expand vertically to accommodate more entries, and all other data below it needs to move accordingly. When the locations of data fields vary from one form to another, this form type are called “semi-structured”. Otherwise, the forms are called “structured” if data fields always have fixed relative locations on a page.

Form-like documents exhibit a high layout variation. The layout design can be affected by culture, customs, language, and aesthetic considerations. For example, we found Japanese forms use more colors and nested table headers than US forms. Since CJK characters can be written vertically, the text orientation in cells of a Japanese form can be both horizontal and vertical.

3 aiDocuDroid — smart template-based form information extraction

The overall IE workflow of aiDocuDroid (Fig. 1) contains two parallel paths. The upper path is form registration that defines the form type and identifies various form regions and field associations. The form templates are stored in a library to be matched with other input forms. The lower path is form processing that automatically identifies the type of an input form and then extracts structured data from it and stores them to a database.

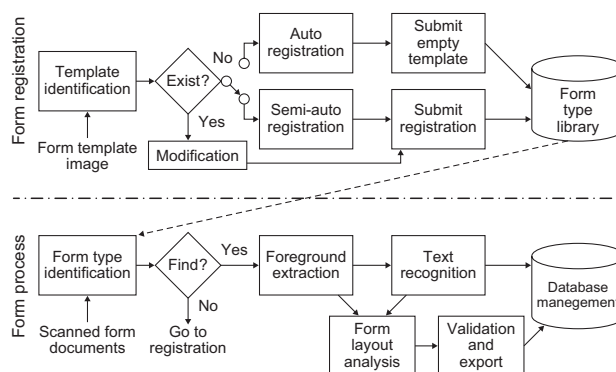


Fig. 1 Workflow of information extraction using aiDocuDroid.

3.1 Form registration

Form registration is the process of creating a template which defines a new form type to which any forms belong can be extracted by the system. A template specifies region types and their spatial arrangement, including 1) key texts and corresponding content fields, and table headers and entries; 2) regions of plain texts; 3) regions of selection elements such as checkboxes; and 4) regions excluded from extraction.

Our form registration requires a minimal amount of user guidance and is robust enough to deduce the rest. The registration consists of two steps. The first step is identification of region types, drawn by the users (Fig. 2(a), Fig. 3(a)); the second step is automatic field label association and table structure detection from the previous user-drawn regions. Users can verify the generated templates and renaming detected key (rekeying) using the form registration user interface (Fig. 2(b), Fig. 3(b)).

Templates can be created from either filled or empty forms. If the form is empty (auto registration in Fig. 1), users only need to draw regions of exclusion (Fig. 2(a)), with the rest such as field labels, table headers, as well as the associated content regions being detected automatically (Fig. 2(b)).

If a filled form is used for template creation (semi-auto registration in Fig. 1), users also need to draw the regions of field labels such as table headers and regions of plain texts in addition to regions of exclusion. A single bounding box can be drawn to enclose all aligned field labels, such as those in a table header even without column delimiting lines. These field labels will be separated during automatic detection. In Fig. 3(b), each detected field label has an arrow indicating the relative location of associated contents. For tables, the detected table header can associate any number of entries below. This gives our solution the robustness in extracting tables that may expand or shrink.

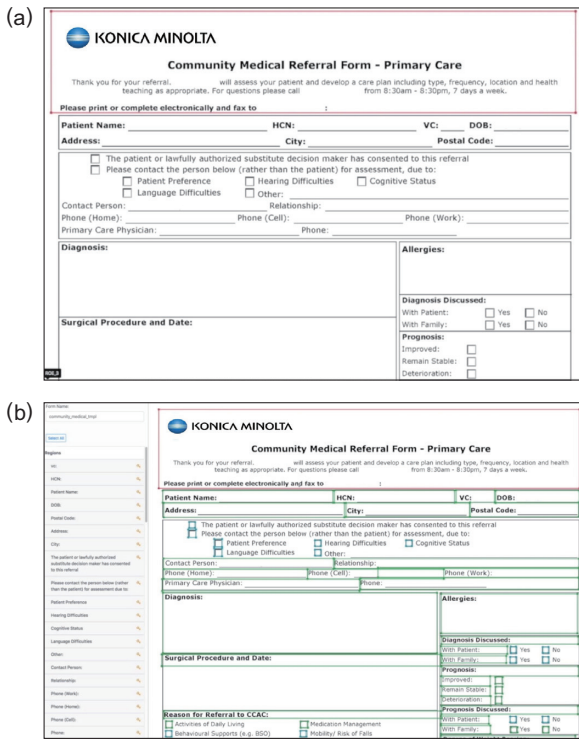


Fig. 2 Form registration using unfilled form. (a) User draws a region of exclusion (red). (b) Automatic field label association and table structure detection. Field labels are recognized and ready for user verification and rekeying.

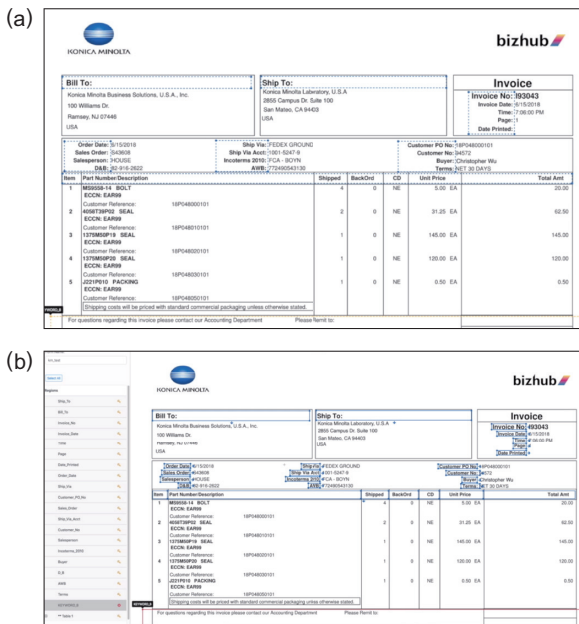


Fig. 3 Registration using filled form. (a) User draws regions of field labels (blue dashed) and plain text regions (yellow dashed). (b) Automatic field label association and table structure detection. Different from unfilled forms, each detected key region indicates by an arrow the relative location of its associated text.

In the form registration user interface, users can set customized rules such as regular expressions or formulas to assert field relationships, for example, values in the last table column should sum up to the total, to verify the extracted results.

3.2 Form Processing

Form processing is the process of information extraction, which involves form type identification, foreground extraction, text recognition, and layout analysis (bottom path in Fig. 1).

3.2.1 Form type identification

Form type identification is to match an input form to one of the templates registered in the form type library. During this step, the input form image is analyzed and features on the form are determined. The form type library is searched for the closest match based on the input feature set. A match is found if the discrepancy is within a certain threshold. The search is formulated as a graph matching problem with a great improvement in accuracy, efficiency and robustness under significant scale change and image noise.

3.2.2 Foreground extraction

Foreground extraction is to separate form foreground pixels such as text, lines, table structure, logo, checkboxes, etc. from the background pixels. This step utilizes AI technologies to eliminate the color variation in backgrounds and text, and different types of image artifacts introduced during scanning and image compression. We developed Deep Learning models that are trained with millions of samples synthesized from real form images.

3.2.3 Text recognition

Text recognition also employs Deep Learning to convert the field images to text strings, field by field. A field may be extracted by multiple images if its texts span multiple lines or are widely separated. Our latest AI-based OCR technology enables the system to achieve high recognition accuracy for both printed and handwritten texts in English and Japanese.

3.2.4 Form layout analysis

Layout analysis is the last step of form information extraction. It tries to reconstruct the logical relationship among entities of input forms by utilizing results from foreground extraction and text recognition, as well as information from templates. Spatial information is deduced from the detected lines and table structures from the foreground extraction, while textual and semantic information is obtained from recognized field texts. Matched with the logical structure defined in the template, aiDocuDroid can accurately extract the required data with its original semantic relationships.

3.3 Validation tool

aiDocuDroid’s validation tool provides users with a side-by-side view to verify the extraction results. The original form image is shown on the left and an output panel is shown on the right. Users can hover the cursor over any field text in the output panel to see the corresponding field image. Table format is retained in the output panel. Users can also use voice guided proofreading to streamline the verification process. As mentioned in form registration, customized rules set up during form registration can be used to verify asserted relationships among numeric values.

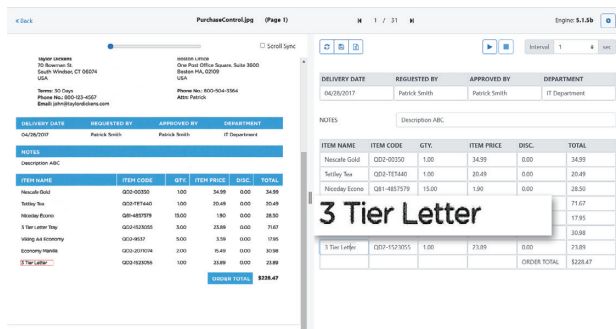


Fig. 4 Validation tool of aiDocuDroid.

4 Other IE solutions

In this section we focus on three representative IE solutions: Amazon’s Textract⁵⁾ (template-free, fully automated), Microsoft Azure Form Recognizer⁶⁾ (template-free, semi-automated) and DocParser⁷⁾ (template-based, semi-automated).

4.1 Amazon Textract

Amazon Textract is an AI-based cloud service that automatically extracts text and data from forms. For end users, Textract is easy-to-use — no manually configured templates are required. For developers, they don’t need to update any customized code for a specific form type when its format or layout changes. These benefits are due to its purely data-driven approach. Textract has been trained on millions of document images across many different industries. It can detect key-value pairs and preserve composition of data stored in tables. But it currently only supports extraction of documents in printed English.

Our evaluation shows Textract can correctly extract the majority of key-value pairs as well as detecting checkboxes and linking its keys. However, its behavior is not very consistent. For key-value pairs extraction, less errors are made when the value is at the right of the keys, but it fails more often when the

content text is under the key text (Fig. 5(a)), or text is rotated to be vertical (Fig. 5(b)). It can also mistake key value pairs in a tabular layout for tables, with key texts extracted as contents (Fig. 5(c)).

- (a) **CORPORATE OFFICE**
75 Maxess Road
Melville, NY 11747-3151
(516) 812-2000
- (b)

Residency	City	Season	Gender
- (c)

Parents’ Contribution	\$
Grants – specify _____	\$
Scholarships – specify _____	\$
Student Employment Income	\$
Total Projected Contribution	\$

Fig. 5 Some cases that Textract fails to extract the data correctly. (a) Value text under the key text. (b) Text rotated to be vertical. (c) Key value pairs in a tabular arrangement. Textract identifies the left columns as contents instead of keys associated with contents in the right column.

Textract can correctly extract tables when there are ruling lines separating table rows and columns. It sometimes succeeds in extracting tables without row/column separators (Fig. 6(a)) but fails on others (Fig. 6(b) and Fig. 6(c)). It may also fail when the key text spans multiple lines in the table header (Fig. 6(d)).

- (a)

Billing Item Description	Units	Rate Per Unit	Flat Rate	Amount
EE Self Service	3	0.2500	0.00	0.75
Base Payroll Charge	0	0.0000	13.00	13.00
- (b)

#	Product name	Delivery	Qty.	Price
1	Parallels Desktop for Mac Pro Edition Upgrade (1 Year)	electronic	1	\$49.99
- (c)

Name of School	Location	Dates Attended	Degree Received
USC	San Bruno	10-15-2003	Bachelor's
UCLA	Westwood	08-14-2016	Master's
UCSB	Santa Barbara	02/25/2019	Ph.D
- (d)

ITEM NO.	QUANTITY ORDERED	PART NUMBER AND DESCRIPTION	PRICE	UNIT	BACK ORDER	SHIPPED	AMOUNT
		Ultimate Destination					

Fig. 6 Various table structures that Textract inconsistently extracts. (a) No row or column separators, header boxed. (b) No row or column separators, header in bold. (c) No column separators, header in bold. (d) Multi-line texts in table headers.

For common structured forms, Textract is a great solution for end users if no document classification is needed. Otherwise, Textract may fail to extract table entries etc., so further manual correction is necessary to ensure the integrity of extracted information.

4.2 Microsoft Azure Form Recognizer

Form Recognizer is also a cloud service that employs machine learning to identify and extract structured information from forms. Different from Textract, Form Recognizer requires users to train their customized machine learning model on their own data if their documents are not sales receipts or business cards for which pre-built models are available.

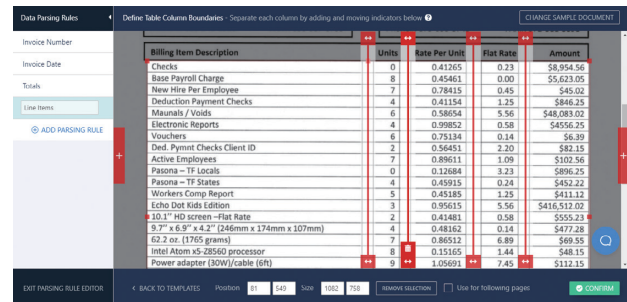
To develop their own model, users need to provide five input forms for training and need to test the trained model and retrain it if the model does not satisfy the requirement. Users are encouraged to first train without ground truth labels, where Form Recognizer uses unsupervised learning to understand the layout and relationships between fields and entries. If extraction quality is not good enough, users can further label their data and train a model using supervised learning. Since our goal is to evaluate template-free approaches that do not require much user input, we focus on the first approach (unsupervised training with unlabeled data).

We tested seven different form types. For each type, there are multiple forms with varying contents and structure. We test a combination of these scenarios: training with and without layout variations; testing with the same or different layout as training samples. Overall, Form Recognizer's customized machine learning model shows certain capability in extracting key-value pairs, especially when the key and value are separated by colons. However, missing key-value pairs, misidentification of value as keyword and vice versa, as well as incorrect key value association are very common when form structure or layout changes such as expandable tables. Among the combination of train/test scenarios, only training without layout variation and testing with the same layout gives slightly better results compared to other scenarios. When trained using unfilled forms, the model shows no significant improvements.

4.3 DocParser

DocParser represents the traditional template-based document processing software using zonal OCR. It can extract data from forms with unknown layouts thanks to manually created templates. To ease the template creation (called "parsing rule creation" in its documentation), users can load presets of popular parsing rules for common data fields. Otherwise, users need to create one parsing rule for each data field in a form. For tables, one parsing rule is needed, but users have to specify column separator

and table boundary (Fig. 7). For common document types such as invoice and purchase order, DocParser can automatically create popular parsing rules (date, totals etc.) for that document type to reduce the manual effort. Building one parsing rule can take from one minute to several minutes depending on the complexity of the data and user's familiarity of the software.



Billing Item Description	Units	Rate Per Unit	Flat Rate	Amount
Checks	0	0.41205	0.23	\$8,954.56
Base Payroll Charge	8	0.45461	0.00	\$5,623.05
New Hire Per Employee	7	0.78415	0.45	\$45.02
Deduction Payment Checks	4	0.41154	1.25	\$846.25
Manuals / Voids	6	0.58654	5.56	\$48,083.02
Electronic Reports	4	0.99852	0.58	\$4556.25
Vouchers	6	0.75134	0.14	\$6.39
Decl. Payment Checks Client ID	2	0.56451	2.20	\$82.15
Active Employees	7	0.88611	1.09	\$102.56
Passma - TF Locals	0	0.12684	3.23	\$896.25
Passma - TF States	4	0.45915	0.24	\$452.22
Workers Comp Report	5	0.45185	1.25	\$411.12
Echo Dot Kids Edition	3	0.95615	5.56	\$416,512.02
10.1" HD screen - Flat Rate	2	0.41481	0.58	\$555.23
9.7" x 6.9" x 4.2" (246mm x 174mm x 107mm)	4	0.48162	0.14	\$477.28
62.2 oz. (1765 grams)	7	0.86512	6.89	\$69.55
Intel Atom X5-28560 processor	8	0.15165	1.44	\$48.15
Power adapter (30W) cable (6ft)	9	1.05991	7.45	\$112.15

Fig. 7 Creating parsing rules for a table requires the user to identify the delimiting lines.

For tables that may expand or shrink, DocParser cannot accurately extract the table data if a single template is used. Users need to create multiple templates for each table layout even if they belong to the same form type. The same situation also applies to key-value pairs: if their positions vary within a document type, DocParser needs different templates to extract them, which further increases the manual effort from users.

5 Discussion

At present, there are two stages that require manual work when using an IE software. The first stage is at the beginning of the process, where users provide some guidance in the form of a template. For the data-driven approach, this part may involve no manual effort (Amazon Textract) or creating a customized machine learning model by the users (Microsoft Azure Form Recognizer). The other stage is at the end of the process, where users verify and correct extracted data, regardless of using templates or not. Although template-free approach can work well for forms with similar structure and layout used for training, it may omit large portions of data silently when form structures deviate from their training distribution. This requires a huge amount of manual correction. Even though there is no manual effort spent in template creation, the overall manual effort required by a template-free solution may be more than that of a template-based one.

Compared with template-based solutions such as DocParser, aiDocuDroid requires less user effort in creating form templates. For the example of creating parsing rules for a table, users do nothing (for unfilled template form) or draw only the region of table header (for filled template form). aiDocuDroid can automatically separate table rows and columns and find the table boundaries.

For the samples tested, our template-based approach involves less manual effort with more accurate and complete the extraction results than other solutions. This shows that our balanced approach can achieve the best accuracy across diverse form layouts. That's why we keep using templates, but smartly take away those tedious parts, giving users a fluid experience.

Reference

- 1) Grand View Research <https://www.bloomberg.com/press-releases/2019-07-16/ocr-market-size-worth-13-38-billion-by-2025-cagr-13-7-grand-view-research-inc>
- 2) Forms Processing - user experiences of text and handwriting recognition (OCR/ICR), AIIM 2012 www.aiim.org / Parascript LLC 2012 www.parascript.com
- 3) Smith, Ray. "An overview of the Tesseract OCR engine". Ninth international conference on document analysis and recognition (ICDAR 2007). Vol. 2. IEEE, (2007).
- 4) Majumder, Bodhisattwa Prasad, et al. "Representation Learning for Information Extraction from Form-like Documents". Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. (2020).
- 5) <https://docs.aws.amazon.com/textract/latest/dg/what-is.html>
- 6) <https://docs.microsoft.com/en-us/azure/cognitive-services/form-recognizer/overview>
- 7) <https://support.docparser.com/>